HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY

FACULTY OF COMPUTER SCIENCE AND ENGINEERING
COURSE: DATABASE SYSTEM LAB (CO2014)

Report

SCHOOL DATABASE MANAGEMENT SYSTEM

Class: CC03
Group: 08

Advisor:	Đỗ Thanh Thái	
Student:	Lý Thụ Phong	2252616
	Nguyễn Quang Duy	2252120
	Trần Duy Đức Huy	2252263

Ho Chi Minh City, October 23rd 2024

Table of Contents

1 Introduction

In modern schools, managing student data, academic performance, and administration efficiently is essential to support high-quality education. This report describes a school database system designed to handle key information about students, courses, teachers, classrooms, and exams. Built using Oracle SQL, the database automates data storage and retrieval, making it easier to track student progress, manage course enrollments, and maintain records.

Oracle SQL was chosen for its strong support of relational data, scalability, and security. The setup involved configuring the system to handle large volumes of data smoothly, with steps like optimizing memory and storage, setting up tablespaces, and enabling security features like user authentication and role-based access. This ensures that data access is limited according to user roles (administrators, teachers, and students) keeping information secure and organized.

1.1 Objective

The main goal of this database is to make school data management more efficient. It provides a well-organized system to handle student records, course details, teacher assignments, and exam results. This system replaces manual data handling, improves accuracy, and enhances access control and security.

1.2 Scope

The database covers all stages of a student's academic journey, from enrollment and course registration to tracking exam performance. It also includes student health records, parent information, classroom scheduling, and materials management, bringing together all key information for school stakeholders.

1.3 Outline

This report will detail the design and implementation of the school database system. It begins with an explanation of the entity-relationship model, followed by the mapping of the ER schema to a relational database schema. We will then discuss the constraints applied to ensure data integrity, SQL implementation, and security considerations. Finally, the report will conclude with an overview of potential future enhancements to the system.

2 Entity-Relationship Model Design

The Entity-Relationship Model of the school database serves as the foundation of the system. It visually represents the various entities in the database and the relationships

between them. This model illustrates how the data is structured, allowing for a clear understanding of how different parts of the system interact.

2.1 Strong Entity Types and Linking Relationships

The school database is built around several strong entities, each with its own attributes and relationships.

Table 1: *List of Strong Entities*

Entity	Attributes	Description
Student	**Personal Information:** Student_ID, Lname, Fname, Gender, DoB. **Contact:** Email, Phone Number, Address. **Performance:** GPA, Status, Enrollment Year	A person who enrolls in courses and participates in exams.
Teacher	**Personal Information:** Teacher_ID, Lname, Fname, Gender, DoB. **Contact:** Email, Phone Number, Address. **Employment:** Years of Experience, Specialization-Subjects.	A person responsible for teaching one or more courses.Teachers belong to departments and have specific expertise.
Classroom	**Classroom_ID:** Building, Room. **Availability:** Date, Time **Capacity:** Number of Students	A physical space where courses are conducted. It can be assigned to one or more courses.
Course	Course_ID, Course_Name, Course_Description	A subject or topic being taught over a specified period.
Exam	Exam_ID, Exam_Date, Exam_Type	An assessment tool used to evaluate a student's understanding and performance in a particular course.
Parent	**Personal Information:** Parent_ID, Lname, Fname, DoB, Relationship **Contact:** Email, Phone Number, Address	A parent or a guardian of students.

These strong entities are linked together through various relationships, such as students enrolling in courses, teachers teaching courses, and students taking exams. These relationships form the backbone of the database and ensure that data is consistently connected and accessible.

Table 2: *List of Linking Relationships*

Relationship	**Cardinality**	**Attributes**	**Description**
Student takes Exam	Many to Many	Grade, Score, Status	A student can take multiple exams; an exam can be taken by multiple students. When a student takes the exam, he/she will receive the grade and score of that exam.
Student attends Class of Course	Many to One to Many	None	A student can attend multiple classes, and each class is associated with a specific course. Likewise, each course can have multiple classes, and each class can have multiple students.
Student has Parent	Many to Many	None	A student can have many parents or guardians; A parent can be the guardian of many students.
Student has health record	One to Many	Record_ID, Allergy, Medical_Note	A student can have multiple health records.
Teacher teaches Class	One to Many	None	A teacher can teach multiple classes; a class can only be taught by one teacher.
Teacher grades Exam	One to Many	Grading_Date	A teacher can grade multiple exams. And the date that the

			teacher grades the exam will be tracked.
Course conducts Exam	One to Many	None	Multiple exams can be conducted by one course.
Course has Prerequisite	Many to Many	None	A course can have multiple prerequisites; a single course can be a prerequisite for multiple courses.
Course includes class	One to Many	Class_ID	A course can include many classes; A class can only be associated to one course.
Class schedules to classroom	Many to Many	Day_of_week, Start_time, End_time	A course can be schedules to multiple classrooms; a classroom can host multiple courses.

2.2 Weak Entity Types and Corresponding Strong Entities

Some entities in the database depend on the existence of a strong entity and cannot exist on their own. These are known as weak entities.

2.2.1 Health Record

- **Corresponding Strong Entities:** Student
- **Reason for weakness:** The Health Record entity depends when a student is in the system for its existence. A Health Record is meaningful when it is tied to a specific student.
- **Attributes:**
  - Record_ID: The partial key that, combined with the primary key of the Student, uniquely identifies each Health Record.
  - Allergy: Information about any allergies that the student may have.
  - Medical_Note: Additional medical information about the student, such as medical conditions, chronic diseases, treatment notes.
- **Primary Key:**
  - Record_ID

    - Student_ID (from the strong entity Student)
- **Description:** The Health Record entity stores the medical information of a student, including allergies, and medical notes. Since it depends on the student, it is classified as a weak entity, identified using a composite key that includes both the Record_ID and Student_ID.

2.2.2 Material

- **Corresponding Strong Entity:** Course
- **Reason for Weakness:** The Material entity depends on a specific course for its existence. A material (such as lecture slides, assignments, or reading lists) is only meaningful when it is tied to a particular course.
- **Attributes:**
    - Material_ID: The partial key that, combined with the primary key of the Course, uniquely identifies each material.
    - Type: The type of material (e.g., "Lecture Slides", "Assignment", "Textbook").
    - Title: The title or name of the material (e.g., "Week 1 Lecture Slides", "Homework 3").
- **Primary Key:**
    - Material_ID
    - Course_ID (from the strong entity Course)
- **Description:** The Material entity stores information about the different materials that are part of a course, such as assignments, lecture notes, and reading materials. Since it depends on the existence of a specific course for its identification, the Material entity is weak and identified using a composite key that includes both the Material_ID and the Course_ID.

2.2.3 Class

- **Corresponding Strong Entity:** Course
- **Reason for Weakness:** The Class entity is dependent on a specific Course for its existence. A Class (such as a specific section or instance of a course that takes place during a particular semester or term) cannot exist without being associated with a Course.
- **Attributes:**
    - Class_ID: The partial key that, when combined with the primary key of the Course, uniquely identifies each class.

- **Primary Key**:
  - Class_ID
  - Course_ID (from the strong entity Course)
- **Description:** The Class entity represents specific sections or offerings of a Course. Each Class is linked to exactly one Course, and a Course may have multiple Classes. Since the Class entity depends on the existence of a Course for its identification, it is considered a weak entity. The Class_ID by itself is not enough to uniquely identify a Class; it must be combined with the Course_ID from the associated Course.

2.3 Key Attributes and Descriptions

The key attributes in each entity shape the structure of the school database, ensuring that all essential data is accurately captured and easily accessible. Each attribute was chosen to support data integrity and the system's overall functionality.

2.3.1 Student

Attribute	Description
student_id	A unique identifier for each student. This serves as the primary key in the Student table and is used to reference the student in other tables.
fname	The first name of the student.
lname	The last name of the student.
gender	A single-character field to denote the gender of the student, constrained to 'M' (male) or 'F' (female).
dob	The date of birth of the student, used for age calculations and validation of enrollment.
email	A unique email address for each student, used for communication. This field is required and has a unique constraint.
phone_number	A unique phone number for each student, used for contact purposes.

address	The home address of the student. This field is optional but useful for administrative purposes.
gpa	The student's grade point average, constrained to a value between 0.0 and 4.0.
status	The current status of the student (e.g., 'active', 'inactive', 'graduated', 'suspended'), ensuring clear tracking of academic progress
enrollment_year	The year the student enrolled, with a check constraint to ensure it aligns with the student's age.

2.3.2 HealthRecord

Attribute	Description
student_id	A foreign key linking the health record to a specific student.
record_id	A unique identifier for each health record, used as the primary key.
medical_note	A description of any medical issues or health history for the student.
allergy	Any known allergies the student has, aiding in health management.

2.3.3 Parent

Attribute	Description
parent_id	A unique identifier for each parent, serving as the primary key.
fname	The first name of the parent.
lname	The last name of the parent.

gender	A field for the gender of the parent, constrained to 'M' (male) or 'F' (female).
dob	The date of birth of the parent
relationship	The relationship of the parent to the student (e.g., 'Father', 'Mother', 'Guardian').
email	A unique email for the parent, used for contact purposes.
phone_number	A unique phone number for each parent.
address	The home address of the parent.

2.3.4 Course

Attribute	Description
course_id	A unique identifier for each course, serving as the primary key.
course_name	The name of the course, constrained to be unique.
course_description	A brief description of the course content.

2.3.5 Prerequisite

Attribute	Description
course_id	A foreign key linking the prerequisite to a specific course.
prerequisite_course_id	A foreign key linking to the course that serves as a prerequisite for another course.

2.3.6 Enrolls

Attribute	Description

student_id	A foreign key linking a student to their enrollment in a course.
course_id	A foreign key linking a course to a student's enrollment.

2.3.7 Teacher

Attribute	Description
teacher_id	A unique identifier for each teacher, serving as the primary key.
fname	The first name of the teacher
lname	The last name of the teacher.
gender	A field for the gender of the teacher, constrained to 'M' (male) or 'F' (female).
dob	The date of birth of the teacher
email	A unique email address for each teacher.
phone_number	A unique phone number for each teacher.
address	The home address of the teacher.
years_of_exp	The number of years of teaching experience the teacher has, constrained to be a non-negative integer.

2.3.8 Class

Attribute	Description
course_id	A foreign key linking a course to a specific class.
class_id	A unique identifier for each class, used as a composite primary key with course_id.

teacher_id	A foreign key linking a teacher to the class they are instructing.

2.3.9 Classroom

Attribute	Description
building	The name of the building where the classroom is located, part of the composite primary key.
room	The room number or identifier, part of the composite primary key
capacity	The maximum number of students that can occupy the classroom, constrained to be a positive number.

2.3.10 Attends

Attribute	Description
student_id	A foreign key linking a student to the class they are attending.
course_id	A foreign key linking a course to the attendance record.
class_id	A foreign key linking a class to the attendance record.

2.3.11 Material

Attribute	Description
material_id	A unique identifier for each material serving as the primary key.
course_id	A foreign key linking the material to a specific course.

type	The type of material (e.g., Textbook, Video), constrained to a list of predefined values.
title	The title of the material
upload_date	The date the material was uploaded.
author	The author or creator of the material.

2.3.12 Exam

Attribute	Description
exam_id	A unique identifier for each exam, serving as the primary key.
exam_date	The date the exam takes place.
exam_type	The type of exam (e.g., 'midterm', 'final', 'quiz'), constrained to specific values.
grading_date	The date the exam is graded, constrained to be after the exam date.
teacher_id	A foreign key linking the exam to the teacher responsible for it.
course_id	A foreign key linking the exam to the course it assesses.

2.3.13 Takes

Attribute	Description
student_id	A foreign key linking a student to the exam they have taken.
exam_id	A foreign key linking an exam to the student who took it.
grade	The grade the student received on the exam, constrained to valid grade values (e.g., 'A', 'B', 'C').

score	The numerical score the student received, constrained to be between 0 and 100.
status	The status of the exam, which can be 'Pending', 'Completed', 'Absent', or 'Graded'.

2.3.14 Schedules

Attribute	Description
class_id	A foreign key linking a class to the schedule.
course_id	A foreign key linking a course to the schedule.
room	The room where the class is scheduled.
building	The building where the class is scheduled.

2.3.15 Time

Attribute	Description
class_id	A foreign key linking the time slot to a specific class.
course_id	A foreign key linking the time slot to a specific course.
room	The room where the class is held.
building	The building where the class is held.
day_of_week	The day of the week the class takes place, constrained to valid days (e.g., 'Monday', 'Tuesday').
start_time	The starting time of the class.
end_time	The ending time of the class, constrained to be after the start time.

2.3.16 Specialization

Attribute	Description
teacher_id	A foreign key linking a teacher to their area of specialization.
subject	The subject in which the teacher specializes.

2.4 Entity-Relationship Diagram

The Entity-Relationship Diagram not only illustrates the relationships between various entities in the school database but also serves as a blueprint for how data is connected and flows across the system. After listing all key attributes, the ERD provides a visual structure that defines how these attributes interact between entities and how relationships are maintained.

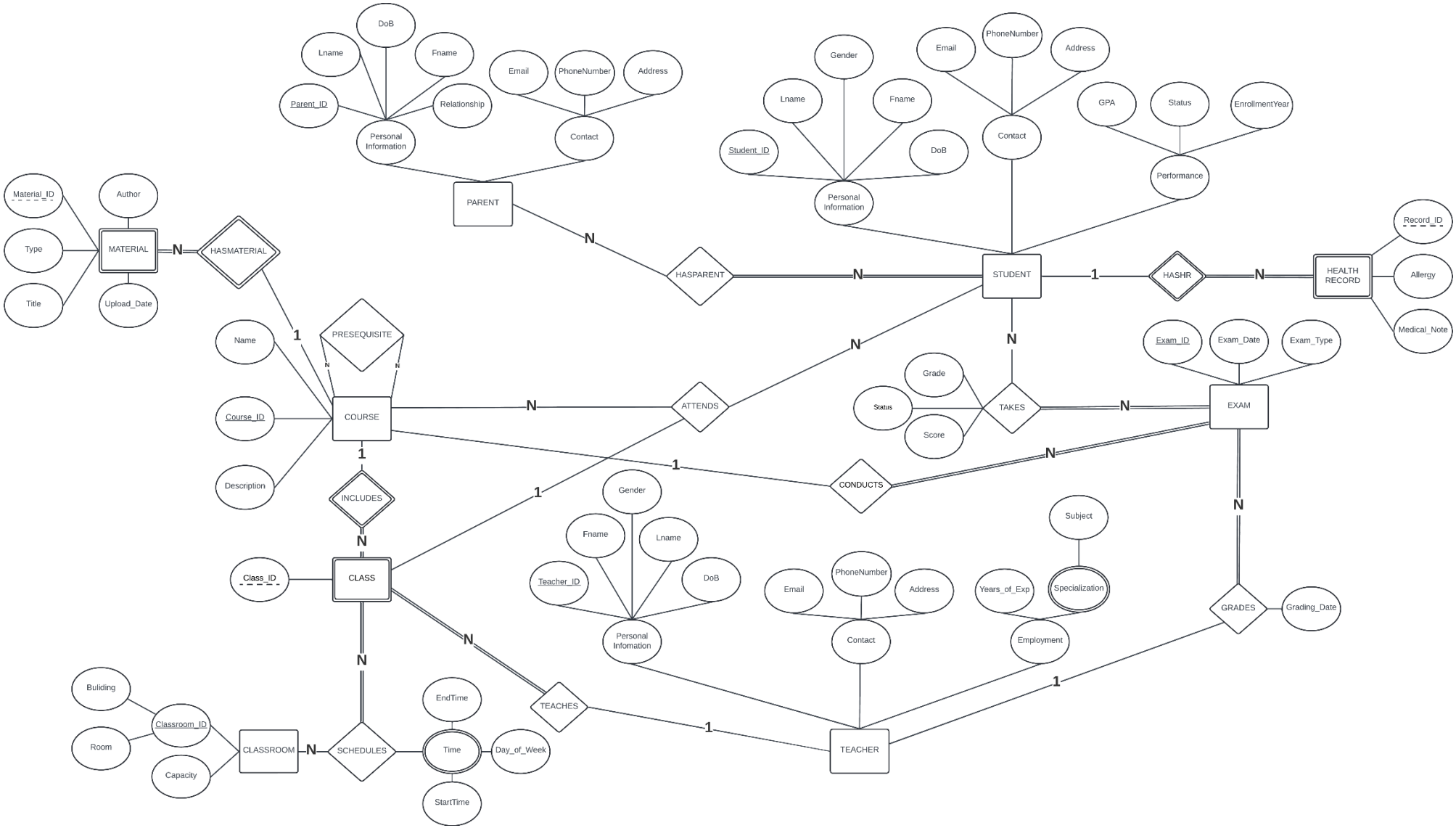

Figure 1: *ERD Diagram*

3 Relational Database Schema

3.1 Mapping to Tables

The relational database schema is derived directly from the ERD. Each entity in the ERD is represented as a table in the relational schema, with its attributes defined as columns. Relationships between entities are mapped using foreign keys and composite keys to maintain the integrity of the data and enforce the rules defined in the ERD.

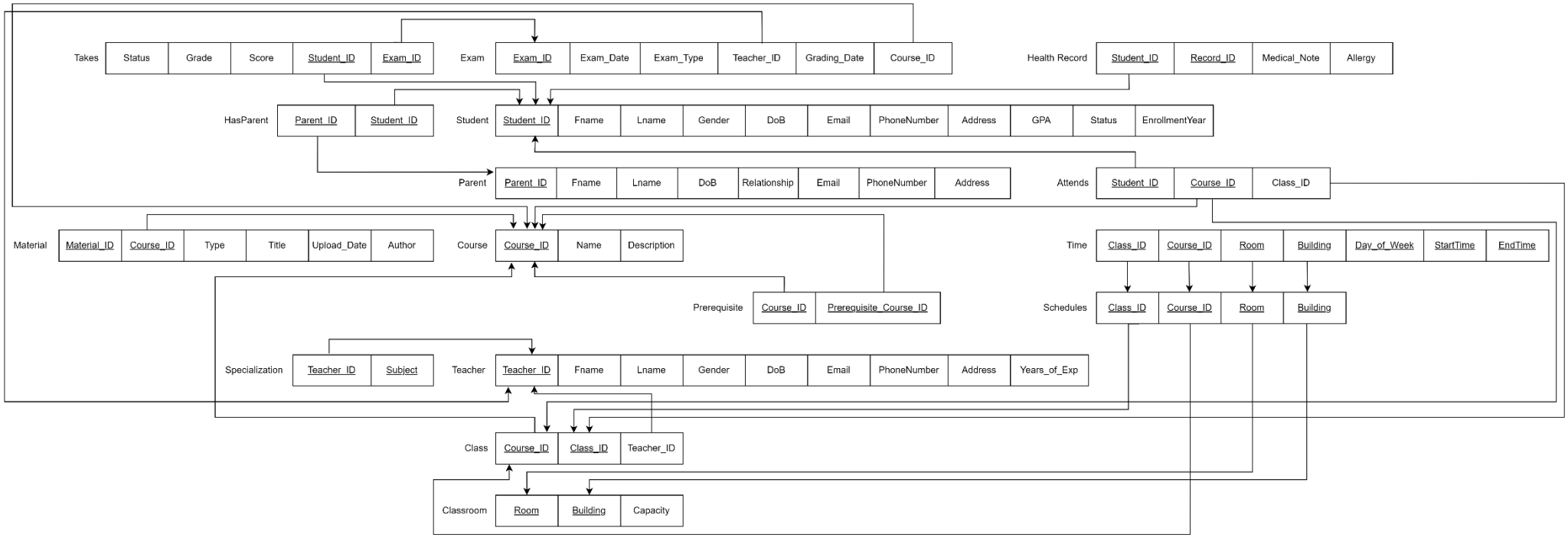

Figure 2: *Relational Data Model*

3.2 Relationships

Our ERD contains one-to-many, many-to-many and ternary relationships but does not include one-to-one relationships. Below, we analyze one example of a one-to-many relationship and one example of a many-to-many relationship. The same principles, such as the use of foreign keys, composite keys, and referential integrity constraints, apply consistently across all similar relationships in the database.

3.2.1 One-to-Many Relationship

A one-to-many relationship exists when a single record in one table corresponds to multiple records in another. In our system, a Teacher can teach multiple Classes, but each class is taught by only one teacher. This relationship is implemented with a foreign key in the Class table, linking it to the primary key of the Teacher table.

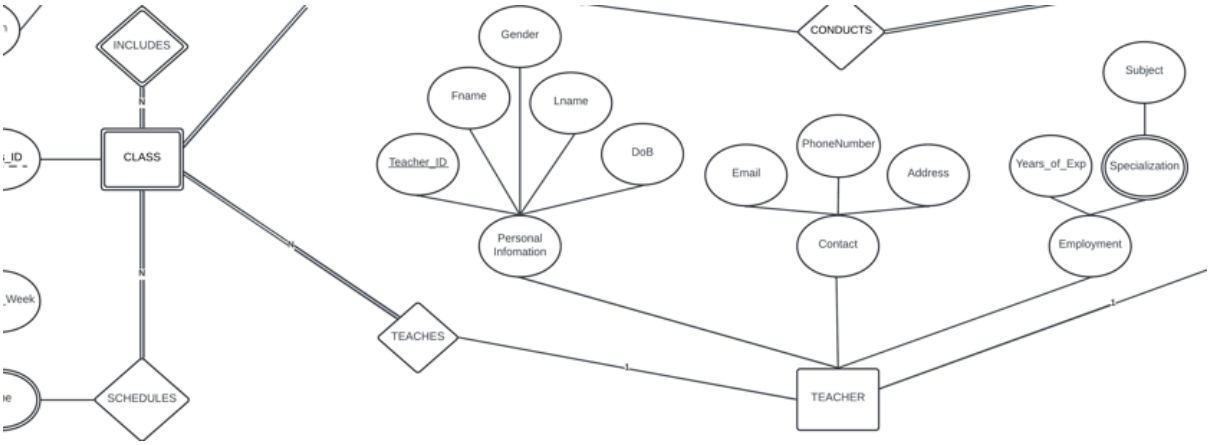

Figure 3: *One-to-Many Relationship between Teacher and Class*

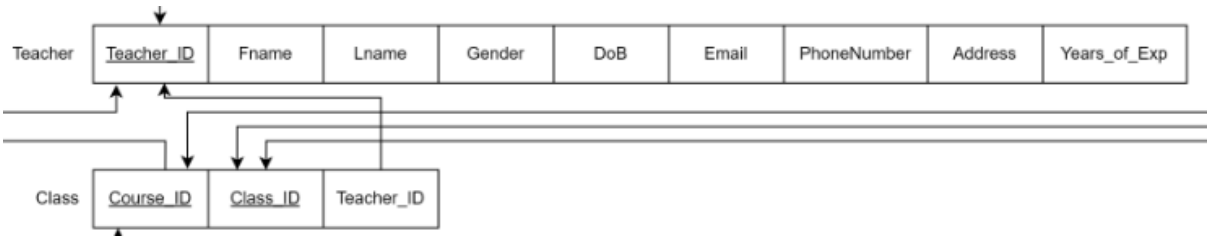

Figure 4: *Relational Data Model of Teacher and Class after Mapping*

The teacher_id foreign key in the Class table establishes the relationship between a teacher and the classes they teach.

3.2.2 Many-to-Many Relationship

A many-to-many relationship occurs when multiple records in one table are linked with multiple records in another. In the school database, the relationship between students

and courses is modeled this way. A student can enroll in several courses, and each course can have many students enrolled. To manage this relationship, a junction table (Enrolls) is used, which links the Student and Course entities. This table contains composite primary keys to ensure the uniqueness of each student-course pair.

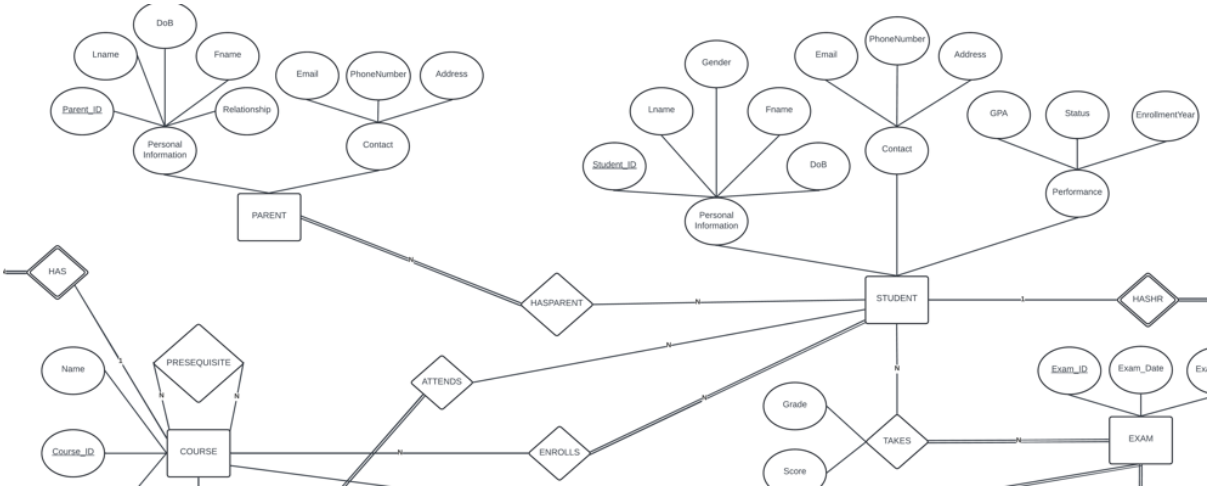

Figure 5: *Many-to-Many Relationship between Student and Course*

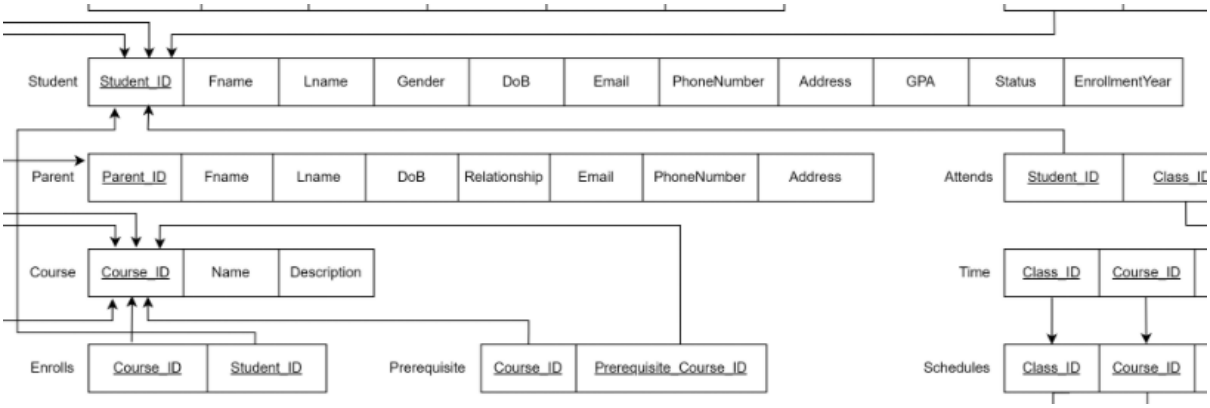

Figure 6: *Relational Data Model of Student and Course after Mapping*

In this example, the Enrolls table acts as the junction table connecting the Student and Course tables. It uses composite primary keys to ensure that each student can enroll in multiple courses and that the relationship is correctly maintained.

3.2.3 Ternary Relationship

A ternary relationship occurs when three different entities are interconnected, allowing for complex interactions among them. In the school database, the relationship between students, classes, and courses is modeled as a ternary relationship. A student can attend multiple classes, each class corresponds to a specific course, and each course can be attended by many students. To manage this relationship, a junction table (Attends) is used,

which links the Student, Class, and Course entities. This table contains composite primary keys to ensure the uniqueness of each student-class-course combination.

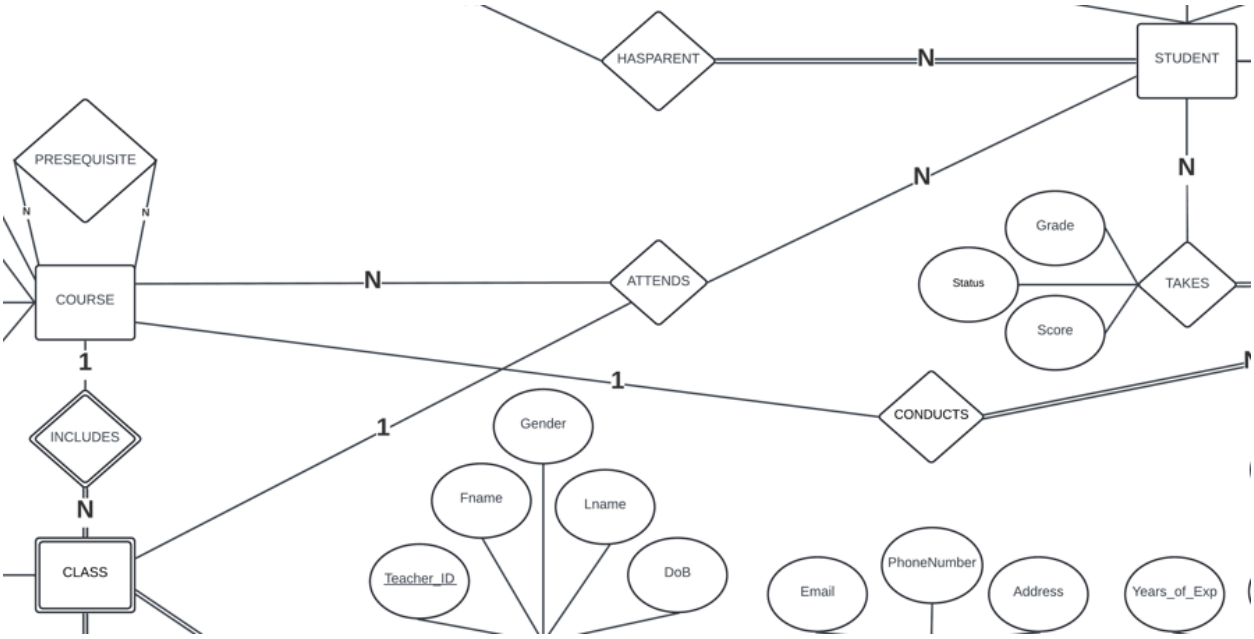

Figure 7: *Ternary Relationship between Student, Class, and Course*

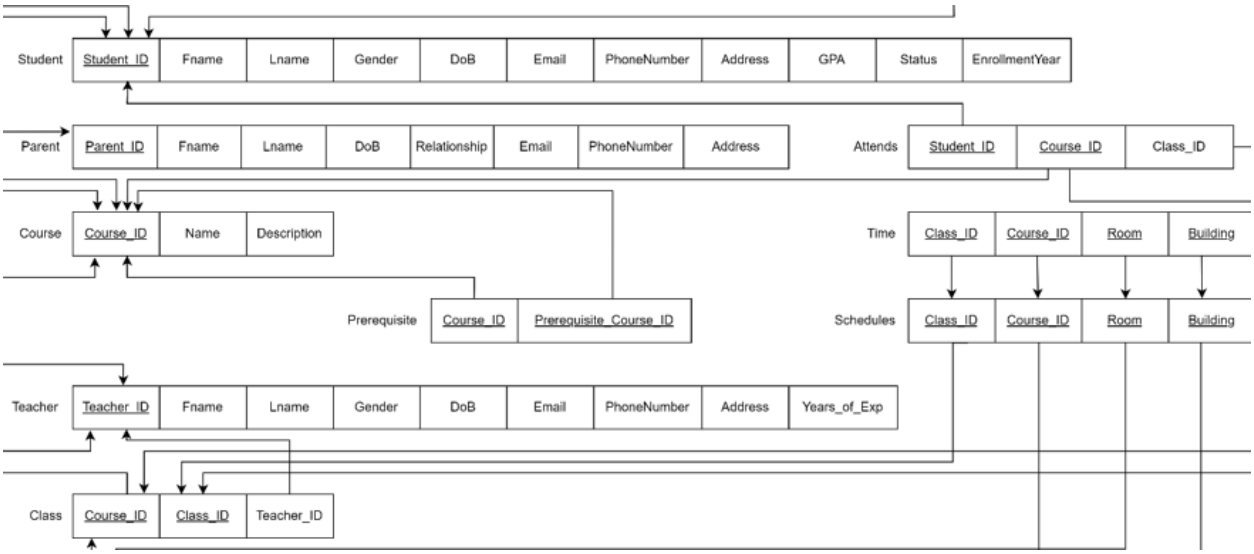

Figure 8: *Relational Data Model of Student, Class, and Course after Mapping*

In this example, the Attends table acts as the junction table connecting the Student, Class, and Course tables. It uses composite primary keys to ensure that each student can attend multiple classes of various courses, while accurately maintaining the relationships between all three entities.

4 Constraints and Validation

4.1 Data Integrity

Data integrity is enforced using various constraints that ensure the consistency, accuracy, and reliability of the stored data. The system applies several types of constraints.

4.1.1 Primary Key Constraints

Primary keys ensure that each record in a table is uniquely identifiable. This prevents duplicate entries and ensures data integrity. A table can have only one primary key, which can consist of a single column or multiple columns (composite primary key). Below are all the primary keys used in our database schema:

Table 3: *Primary Keys*

Table	Primary key	Key type
Student	student_id	Simple Key
HealthRecord	student_id,record_id	Composite Key
Parent	parent_id	Simple Key
HasParent	parent_id, student_id	Composite Key
Course	course_id	Simple Key
Prerequisite	course_id, prerequisite_course_id	Composite Key
Enrolls	course_id, student_id	Composite Key
Teacher	teacher_id	Simple Key
Class	course_id, class_id	Composite Key
Classroom	building, room	Composite Key
Attends	student_id, course_id, class_id	Composite Key
Material	course_id, material_id	Composite Key
Exam	exam_id	Simple Key

Takes	student_id, exam_id	Composite Key
Schedules	class_id, course_id, room, building	Composite Key
Time	class_id, course_id, room, building, day_of_week, start_time, end_time	Composite Key
Specialization	teacher_id, subject	Composite Key

4.1.2 Check Constraints

Check constraints ensure that only valid data is entered into specific fields by restricting the possible values. Below are all the check constraints used in our database schema:

- **Student**:
  - gender must be either 'M' or 'F'
  - gpa must be between 0 and 4.0
  - status must be one of 'active', 'inactive', 'graduated', or 'suspended'
  - enrollment_year must be at least 18 years after the student's birth year
- **Teacher**: years_of_exp must be non-negative
- **Parent**:
  - gender must be either 'M' or 'F'
  - relationship must be 'Father', 'Mother', or 'Guardian'
- **Course**: course_name must be unique
- **Exam**:
  - exam_type must be 'midterm', 'final', or 'quiz'
  - grading_date must be after exam_date
- **Takes**:
  - grade must be one of 'A', 'B', 'C', 'D', 'E', 'F'
  - score must be between 0 and 100
  - status must be one of 'Pending', 'Completed', 'Absent', 'Graded'
- **Material**: type must be one of 'Textbook', 'Video', 'Article', 'Presentation', 'Other'
- **Time**:
  - day_of_week must be one of the seven days
  - end_time must be after start_time
- **General Date and Time Constraints:** Must be in the future or the current date and time

- **DATE:** Stores only the date (without time) in the format YYYY-MM-DD.
- **TIME:** Stores only the time in the format HH:MM:SS.

4.1.3 Unique Constraints

Unique constraints ensure that certain fields contain unique values across all records in the table. The following is a complete list of all the unique constraints used in our database schema:

- **Student**: email and phone_number must be unique
- **Parent**: email and phone_number must be unique
- **Teacher**: email and phone_number must be unique
- **Time**: (room, building, day_of_week, start_time, end_time) must be unique

To enforce the uniqueness of email and phone numbers across Student, Parent, and Teacher, we created a centralized ContactInfo table with unique constraints on both fields. This design ensures each contact detail is stored uniquely and referenced consistently across these entities, promoting data integrity and simplifying contact information management.

4.2 Referential Integrity

Referential integrity ensures that relationships between tables remain consistent by enforcing that all foreign key values must correspond to valid primary key records in the referenced tables. This prevents orphaned records and ensures that all references are meaningful. Below is a complete list of the foreign key relationships in our database schema:

- Student_ID must reference a valid Student_ID in the Student table.
- Exam_ID must reference a valid Exam_ID in the Exam table.
- Course_ID must reference a valid Course_ID in the Course table.
- Parent_ID must reference a valid Parent_ID in the Parent table.
- Teacher_ID must reference a valid Teacher_ID in the Teacher table.
- Classroom (defined by Room and Building) must reference valid Room and Building values in the Classroom table.
- Prerequisite_Course_ID must reference a valid Course_ID in the Course table, ensuring that prerequisites exist for dependent courses.
- HealthRecord must reference a valid Student_ID in the Student table to associate medical records with students.
- Class_ID (along with Course_ID) must reference valid Class_ID and Course_ID in the Class table.

- Enrolls must reference valid Student_ID in the Student table and Course_ID in the Course table.
- Takes must reference valid Student_ID in the Student table and Exam_ID in the Exam table.
- Schedules must reference valid Class_ID and Course_ID from the Class table and Room and Building from the Classroom table.
- Time must reference valid Class_ID, Course_ID, Room, Building from the Schedules table.

5 User Groups and Permissions

In the school database system, different user groups have tailored access to information relevant to their roles. The primary user groups implemented are Admins, Teachers and Students, each with specific views to ensure secure and efficient access to the data they need. This structure protects sensitive information while supporting the operational requirements of each group.

5.1 Admin Permissions and Views

The Admin group has full access to manage the database, oversee user accounts, and ensure data integrity across the system. Admins handle all database operations, making them responsible for system-wide maintenance and user access management.

- Full Database Access: Admins have unrestricted access to all tables and views, including permissions to create, alter, and delete tables and other database objects.
- User and Role Management: Admins can create, update, or delete user accounts, assign roles, and adjust permissions for Teachers and Students, ensuring that access controls are well-maintained.
- Audit and Log Access: Admins can view system logs and perform audits, supporting data integrity, compliance, and troubleshooting within the database.

Admins have the most comprehensive access in the system, enabling them to maintain the integrity and security of the database.

5.2 Teacher Permissions and Views

Teachers have access to student information, course schedules, grades, and materials for the courses they teach. This access is managed through custom views created specifically for each teacher, allowing them to interact with only their assigned data. Here are the main views available to teachers:

- Student Information View: Each teacher can view a list of students in their classes, including essential student information such as names, contact details, GPA, enrollment

status, and course IDs. This view supports teachers in monitoring their students' academic details and managing communication.

- Timetable View: Teachers can view their class schedules, including room assignments, building names, and timings. This ensures teachers have complete information on when and where their classes are held, supporting efficient scheduling and resource management.
- Personal Information View: Teachers have access to their personal information, allowing them to verify and maintain up-to-date records.
- Exam View and Update: Teachers are provided with access to exam details, including exam dates, types, and scores. They can view grades for students in their classes and are authorized to update scores and grades for the exams they administer, enabling them to manage and assess student performance accurately.
- Course Material Management: Teachers can access, add, update, and delete materials related to the courses they teach, ensuring they have the ability to manage instructional resources effectively.
- Specialization View: Each teacher can view their area of specialization, providing visibility into their assigned subjects.

These views provide teachers with the necessary access to efficiently manage their courses and student interactions, while strictly limiting access to only relevant information.

5.3 Student Permissions and Views

Students primarily access information about their academic progress, enrolled courses, and personal schedules. Specific views are created to provide them with the relevant data they need while safeguarding other students' information. The main views available to students are as follows:

- Grade and Score View: Each student can view their own grades and scores for exams they have taken. This view enables students to monitor their academic progress and understand their performance in various courses.
- Health Record View: Students can access their health records, which include medical notes and allergy information. This view ensures that students have access to their own health information as part of the system’s support for student welfare.
- Parent Information View: Students can view contact information and details about their parents or guardians. This view facilitates communication between the student and their family contacts within the system.
- Course View: Students have read-only access to a list of all available courses, along with descriptions, providing them with insight into the subjects available for enrollment.

- Course Material View: Students can access materials for the courses they are enrolled in, including documents, videos, and other resources uploaded by their teachers. This view ensures students have the resources needed to succeed in their courses.
- Timetable View: Each student has access to a timetable showing their class schedule, including course names, room assignments, and timings. This view helps students stay organized and aware of their daily schedules.

These views provide students with access to essential academic and personal information, supporting their educational journey while maintaining strict data privacy controls.

6 Optional Bonus: Audit Implementation for Enhanced Security

To strengthen security and monitor database interactions, an audit mechanism has been implemented as an additional layer of oversight. This audit system captures detailed logs of user actions, tracking any access, insertion, modification, or deletion of data within the database. By auditing activities of key user groups, we ensure accountability and transparency, supporting data integrity and compliance with security standards.

The audit is designed to monitor interactions with tables across the database, specifically for Teachers and Students. The key actions audited include:

- SELECT: Tracks data access to monitor when and how frequently data is viewed.
- INSERT: Logs instances of new data entries.
- UPDATE: Captures any modifications to existing data, which can be critical in tracking changes in sensitive information.
- DELETE: Monitors the removal of records to prevent unauthorized data loss or tampering.

Through this audit implementation, the database system gains a robust tool for monitoring, accountability, and data security, supporting a secure and reliable school management environment.

7 Developing the Database Application

7.1 SQL Statements and Code Repository

This application utilizes various SQL statements to manage and manipulate the school database effectively. The structured relational schema supports operations such as querying student profiles, schedules, and exam records. For the complete SQL code, please refer to the GitHub repository where all relevant SQL scripts and examples are hosted.

GitHub repository: https://github.com/Acht8888/DBMS.git

7.2 Future Development of the Database

The front-end of the application will utilize HTML, CSS, and JavaScript to establish core structure and styling. React.js or Vue.js will be incorporated to enhance interactivity and enable efficient management of features such as student profiles, schedules, and exam records. To achieve a responsive and user-friendly layout, the application will employ Bootstrap or Tailwind CSS.

On the back-end, Node.js with Express.js provides a scalable and secure environment for handling routes, sessions, and user authentication. For data storage, Oracle SQL supports a structured, relational schema. To ensure secure access control, JWT or OAuth will be implemented to restrict data views to authorized users, including admins, teachers, and students. This technology stack supports reliable and efficient school data management.

8 Conclusion

The School Database Management System provides a practical solution for managing data in educational institutions. With a structured design, it organizes information about students, teachers, courses, and other administrative areas, making data easier to access, manage, and protect. The database uses a relational model that supports connections between different types of data, such as student enrollments, course schedules, and exams, ensuring that all data stays organized and linked.

Key features like data integrity checks, user-specific permissions, and audit tracking help keep data accurate and secure, giving each user group access only to what they need. The system is also designed to grow, allowing for future improvements, such as analytics and customized interfaces, to better support students, teachers, and administrators. This database creates a strong foundation for efficient school management and a smoother educational experience.